ФГОБУ ВО «Финансовый университет при Правительстве Российской Федерации»
ВСЕРОССИЙСКАЯ ОЛИМПИАДА ПО ИНФОРМАТИКЕ
«МИССИЯ ВЫПОЛНИМА. ТВОЕ ПРИЗВАНИЕ – ФИНАНСИСТ!»
ОЧНЫЙ ЭТАП, 2025 год

Продолжительность олимпиады – 235 минут. Олимпиадное задание состоит из пяти задач. Для каждой задачи указан ее вес в баллах.
Участник олимпиады самостоятельно определяет последовательность выполнения задач. На одном из языков программирования – C/C++, C#, Visual Basic, Pascal или Python – разработайте *консольные* программы для решения перечисленных ниже задач.

При выполнении задания участник формирует каталог в имени которого указывает свое ФИО. В данном каталоге формирует пять каталогов: Task1; Task2; Task3; Task4; Task5. Решение задачи размещаются в каталоге с соответствующим номером (см. рис П.1**)**

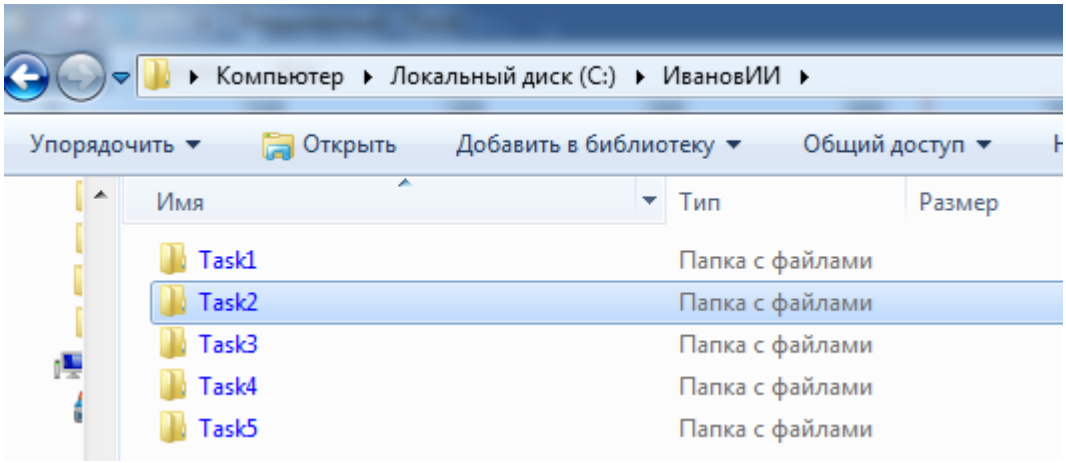


Рисунок П.1 – Структура каталога участника олимпиады

Участник олимпиады должен предоставить членам комиссии на проверку только файлы с исходными текстами программ, которые должны быть названы участником олимпиады в соответствии с выполняемым заданием, например, для языка Python: Task1.py.

Расширение файла должно соответствовать языку. Переименуйте файлы перед сдачей работы, если это необходимо. (см пример на рис. П.2)

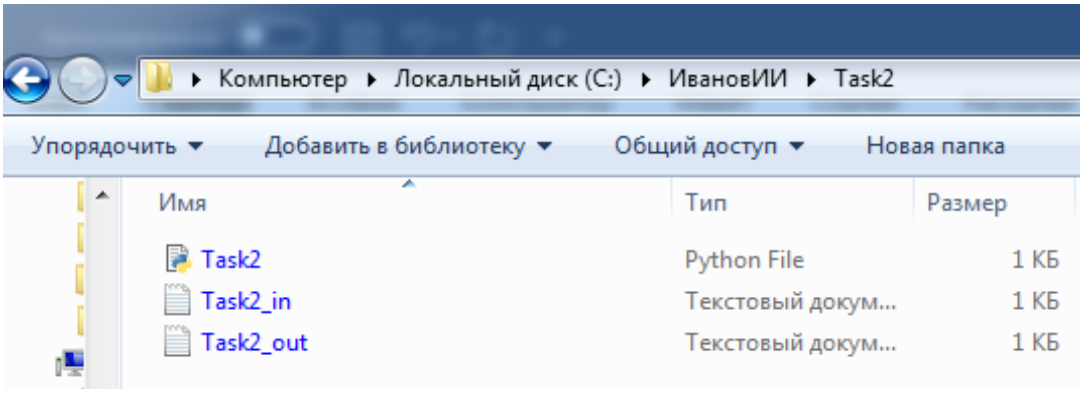


Рисунок П.2 – Размещение файлов в рамках папки задачи

В начале каждой программы должен находиться комментарий с ФИО участника, вариант, номером задачи, языком программирования, средой программирования или

онлайн-компилятора. Например, для С-подобных языков: // Иванов И. И., вариант 1, задача 1, Python 3.7.3, Spyder 3.3.6.

Если файлы с решением задачи, исходных и результирующих данных имеют некорректные названия и/или отсутствует первая строка комментарий, и/или размещены в каталоге участника без учета требований к структуре, то члены комиссии данное решение не оценивают и баллы за решение задания не начисляются.

При решении задач в качестве файлов с исходными данными и выходными данными используется только текстовый файл с расширением *.txt. Если в задаче программной реализации используется файлы с исходными данными и/или выходными данными, то кроме файла с исходным текстом требуется выслать соответствующие файлы. Например, для задания 2 требуется использовать исходные данные из файла, тогда название файла должно быть Task2_in.txt. если в задание 2 требуется сформировать текстовый файл с результатами исполнения программы, то название файла должно быть Task2_out.txt. Число после слова Task соответствует номеру решенного задания, «_in» определяет, что файл с исходными данными, «_out» определяет, что в файле хранятся результаты исполнения кода над исходными данными. Все текстовые файлы с исходными данными создаются участником самостоятельно, в соответствии с представленными в задачах примерами и шаблонами. Ввод и вывод текста осуществляется только латинскими буквами.

По окончанию работы над заданиями участник формирует архив с содержимым решений в соответствии с предложенной выше структурой. Расширение архивного файла должно быть rar или zip (пример см. рис П.3-П.5).

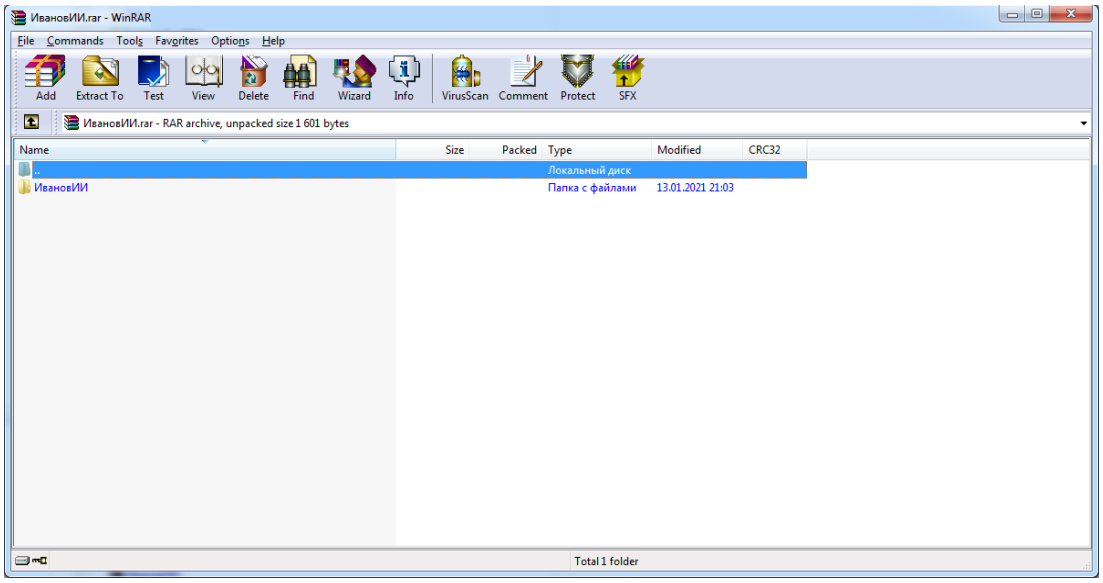


Рисунок П.3 – Пример первого уровня содержимого архива

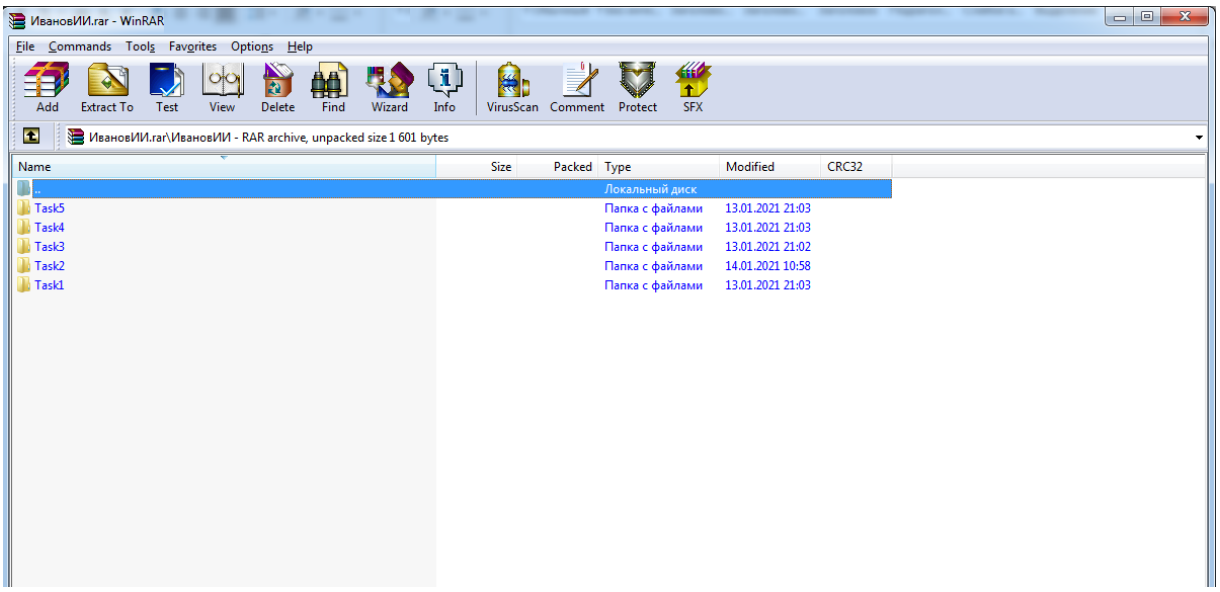


Рисунок П.3 – Пример второго уровня содержимого архива

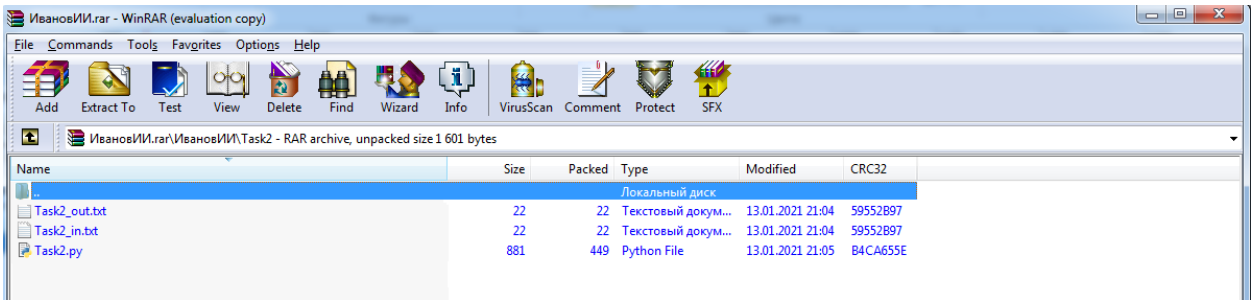


Рисунок П.3 – Пример третьего уровня содержимого архива

Члены комиссии, при проверке заданий, используют операционные системы семейства Windows (версии 7-11). Если члену комиссии не удается запустить файл с исходным кодом на исполнение (например, программа не выполняется в перечисленных ОС и/или не находит файл с исходными данными, и/или не формирует результирующий файл и т. д.), то задание считается не выполненным.

Максимальное количество баллов, которые может набрать участник – 100.

При оценивании решения задачи члены жюри могут снизить баллы за следующие недостатки:

- неполное соответствие решения условию;
- применение неэффективного алгоритма;
- решение задачи только для частного случая;
- отсутствие проверок, приводящих к снижению надежности программы;
- низкое качество интерфейса пользователя;
- несоответствие решения пулу тестовых значений;
- плохая читабельность текста программы и т. д.

При обнаружении использования участником: посторонней помощи в любом проявлении, средств интернет, мобильных устройств и других приемо-передающих устройств, способствующих решению заданий олимпиады, не используя собственные знания, приведут к исключению участника и аннулированию его результатов.

Для программной реализации заданий участник олимпиады должен использовать онлайн-компилятор https://www.onlinegdb.com/, пройдя процедуру регистрации.

При неработоспособности онлайн-компилятора https://www.onlinegdb.com/ участник олимпиады может использовать следующие среды разработки: CodeBlocks, Anaconda, Lazarus, Mono + MonoDevelop, установленные на ЭВМ.

В случае, если программный код участника не запускается в вышеперечисленных средах разработки, комиссия не рассматривает данную работу.

Задания на очный этап олимпиады «Информатика»
Вариант №1

Задание 1 (8 баллов)

Реализовать программу для расчета функции y=y(x), представленной на рисунке 1.1. Значения x вводятся с клавиатуры, результат вычислений выводится на экран ПК.

$$y = log_{10}\left(\frac{\sqrt{9-x^2}}{sin(x)}\right)$$

Рисунок 1.1 — Функция y=y(x)

Задание 2 (12 баллов)

Выполнить программную реализацию расчета значений логической функции четырех переменных (см. рис. 2.1) и вывод результата в файл Task2_out.txt.

$$F = D \rightarrow (A\ and\ B)\ or\ (C\ or\ not\ A)$$

Рисунок 2.1 – Логическая функция четырех переменных

В функции обозначения соответствуют:

- *or* – логическое ИЛИ (дизъюнкция);
- and – логическое И (конъюнкция);
- *not* – логическое отрицание (НЕ);
- → – обратная импликация;

Исходные данные с 16 комбинациями значений переменных требуется считать из файла Task2_in.txt. (пример см. рис. 2.2, порядок столбцов A B C D всегда соответствует данному примеру и не может быть изменен).

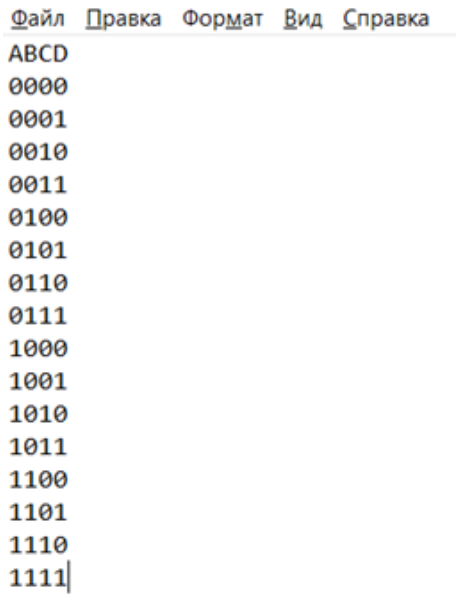
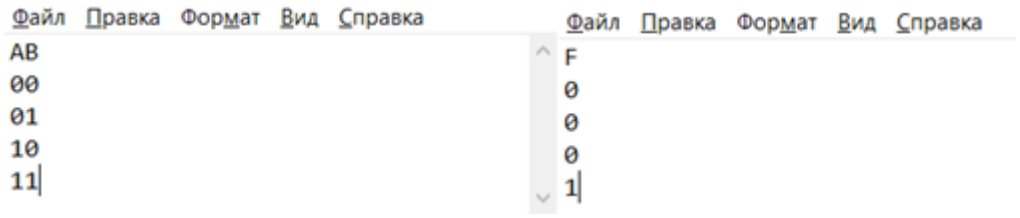


Рисунок 2.2 – Файл Task2_in.txt

Значение F вычисляется разработанной программой (за ручное решение с записью в файл Task2_out.txt баллы не начисляются), и записывается в файл Task2_out.txt.

На рисунках 2.3 a) и 2.3 b) показаны примеры файлов Task2_in.txt и Task2_out.txt, заполненных в соответствии с требованиями, для логической функции от двух переменных F = A and B.



a) b)

Рисунок 2.3 – a) Файл Task2_in.txt, b) Файл Task2_out.txt

Задание 3 (20 баллов)

В переписке между учебными комплексами Финуниверситета для передачи важных сообщений решили использовать шифр. Длина сообщений при этом никогда не превышает 25 символов, включая пробелы.

Процесс шифрования начинается с построения сетки размером 5×5, где каждая ячейка заполняется 25 буквами латинского алфавита (буквы I и J шифруются как I). Каждая строка и столбец сетки задаётся одной из 5 букв: «A», «D», «F», «G» и «X». Заполнение сетки происходит в случайном порядке, поэтому получатель должен знать точное расположение каждого элемента для успешной дешифровки.

	A	**D**	**F**	**G**	**X**
A	F	N	H	E	Q
D	R	D	Z	O	C
F	I	S	A	G	U
G	B	V	K	P	W
X	X	M	Y	T	L

Рисунок 3.1

Рассмотрим процесс шифрования на примере небольшого сообщения: «ATTACK».

На первом шаге каждый символ сообщения заменяется на пару букв, обозначающих строку и столбец соответствующего символа в сетке, так A будет заменено на FF, а B — на GA.

Сообщение: ATTACK.

Шифротекст на первом шаге: FF XG XG FF DX GF

На втором шаге применяется перестановка. Перестановка осуществляется в зависимости от ключевого слова, которое должно быть известно получателю. Пусть в нашем примере таким словом будет «BATTLE». Процесс перестановки заключается в следующем. Вначале создаётся новая сетка, в верхней строке которой записываются буквы ключевого слова. Затем под этим словом построчно записывается, полученный на первом шаге зашифрованный текст.

B	**A**	**T**	**T**	**L**	**E**
F	F	X	G	X	G
F	F	D	X	G	F

Рисунок 3.2

Далее буквы ключевого слова переставляются в алфавитном порядке вместе с соответствующими им столбцами сетки.

A	B	E	L	T	T
F	F	G	X	X	G
F	F	F	G	D	X

Рисунок 3.3

После чего буквы каждого столбца выписываются поочерёдно сверху вниз. Полученная последовательность букв образует окончательный вид шифротекста.

Окончательный вид шифротекста: FFFFGFXGXDGX.

Для восстановления исходного текста необходимо выполнить действия, обратные шифрованию. Обладая ключевым словом, последовательность столбцов можно привести к первоначальному порядку. Зная расположение символов в исходной таблице, можно расшифровать текст.

Необходимо автоматизировать процесс шифрования и дешифрования с помощью описанного метода (пробелы в сообщениях не шифруются и соответственно не дешифруются).

Требования к программе:

1. Пользователь выбирает действие, которое собирается осуществить (реализация меню см. рис. 3.4):

```
Выберите действие:
Шифрование нажмите - 1
Дешифрование нажмите - 2
```

Рисунок 3.4

2. При нажатии «1» программное средство должно выполнить шифрование.

3. При нажатии «2» программное средство должно выполнить дешифрование.

При шифровании в качестве входного файла программа должна использовать файл Task3_in_SH.txt структура которого представлена на рисунке 3.5.

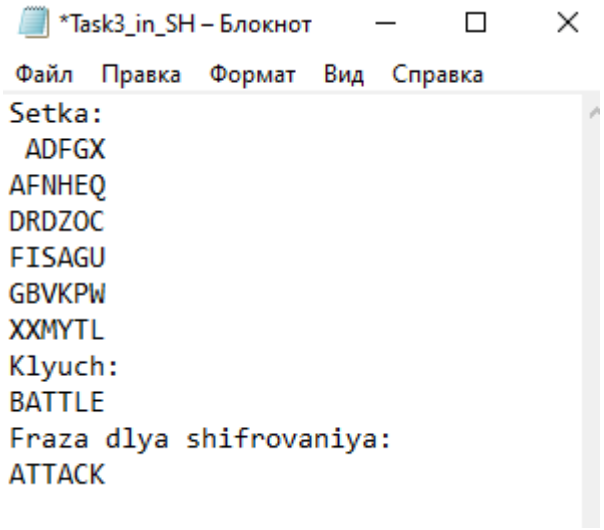


Рисунок 3.5

В качестве выходного файла формируется файл Task3_out_SH.txt структура которого представлена на рисунке 3.6.

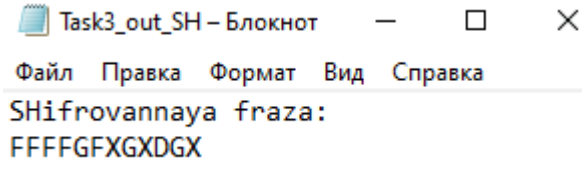


Рисунок 3.6

При дешифровании в качестве входного файла программа должна использовать файл Task3_in_DSH.txt структура которого представлена на рисунке 3.7.

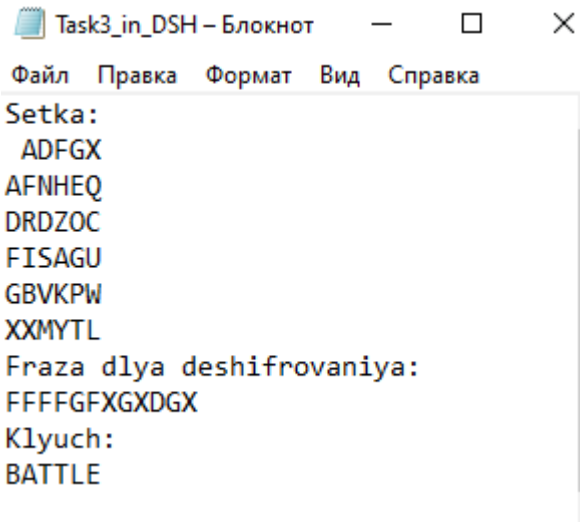


Рисунок 3.7

В качестве выходного файла при дешифровании формируется файл Task3_out_DSH.txt структура которого представлена на рисунке 3.8.

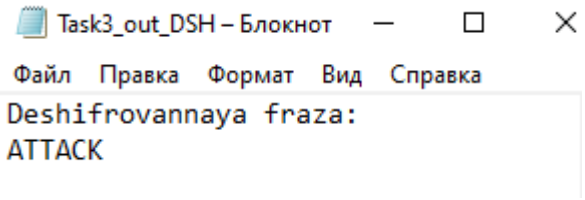


Рисунок 3.8

При шифровании и дешифровании фраз требуется выполнить проверки:

1. На существование в исходном сообщении всех символов, которые будут подвержены шифрованию или дешифрованию (сообщение должно содержать только буквы латинского алфавита и пробелы);
2. Длина сообщения не превышает 25 символов (пробелы из сообщения удаляются или сообщение вводится без пробелов).

При несоответствии требованиям программное средство должно сформировать следующие сообщениях в Task3_out_SH и Task3_out_DSH:

1. «В исходном сообщении указаны символы, которые не могут быть зашифрованы или дешифрованы»;
2. «Длина сообщения не соответствует требованиям».

Задание 4 (25 баллов)

Научная экспедиция проводит многопараметрические измерения климата. Каждый блок данных представлен в виде квадратной матрицы, отражающей некоторую «корреляцию» между различными факторами (температура, влажность, атмосферное давление, солнечная активность и т. п.). Если матрица корреляций симметрична относительно главной диагонали, это говорит о стабильности и предсказуемости системы. Если матрица несимметричная, возможны скрытые факторы или ошибки в экспериментах. Для быстрого анализа необходимо разделить весь массив матриц на две группы: симметричные и несимметричные, сохранив их в разные файлы.

Сформируйте самостоятельно файл Task4_in.txt, в котором хранится k квадратных матриц порядка n. Первая строка файла содержит два натуральных числа k и n, k – количество матриц, n – размерность каждой матрицы ($n \times n$), далее идут k блоков по n строк, в каждой строке ровно n целых чисел. Рисунок 4.1 демонстрирует пример такого файла, в нем $k = 3$, $n = 3$ и всего три матрицы 3 × 3 (все числа условные).

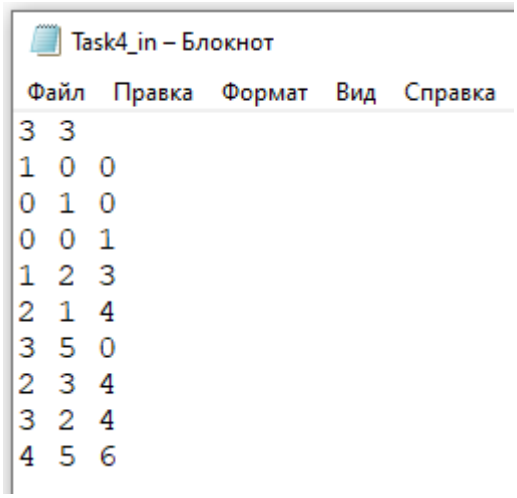


Рисунок 4.1. – Пример файл Task4_in.txt.

Требуется:

1)считать данные из Task4_in.txt;
2) проверить формат (количество строк, столбцов, целочисленность входных значений); если где-то не совпадает число столбцов или встретились нечисловые данные, вывести сообщение об ошибке и завершить программу;
3) убедиться, что $k \geq 1$ и $n \geq 1$, в противном случае — ошибка;
4) для каждой матрицы проверить, является ли она симметричной: $A = (a_{ij})$, $A^T = (a_{ji})$, A симметрична, если $a_{ij} = a_{ji}$ для всех i, j.
5) разделить матрицы на две группы: симметричные и несимметричные.

Выходные данные:

1) файл Task4_1_out.txt хранит все симметричные матрицы: первая строка содержит два числа – количество симметричных матриц k_s и размерность n, далее k_s матриц, каждая по n строк и n столбцов;
2) файл Task4_2_out.txt хранит все несимметричные матрицы: первая строка содержит количество несимметричных матриц k_{ns} и размерность n, затем k_{ns} матриц $n \times n$.

Если какие-то проверки (формат, размерности, целочисленность) не прошли, никакие файлы не формировать/перезаписывать, а вместо этого вывести сообщение об ошибке на экран. Пример результата показан на рисунке 4.2 ($k_s = 2$, $n = 3$ и две матрицы размера 3 × 3) и рисунке 4.3 ($k_{ns} = 3$, $n = 3$ и одна матрица размера 3 × 3).

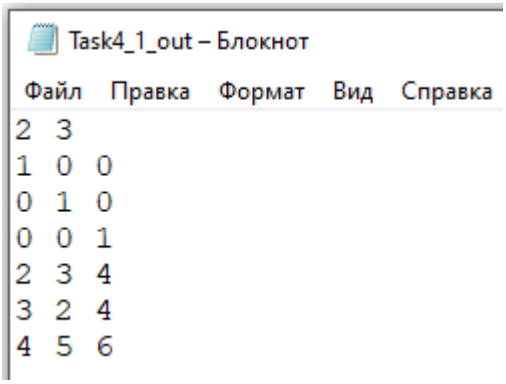


Рисунок 4.2 – Пример файла Task4_ 1_out.txt.

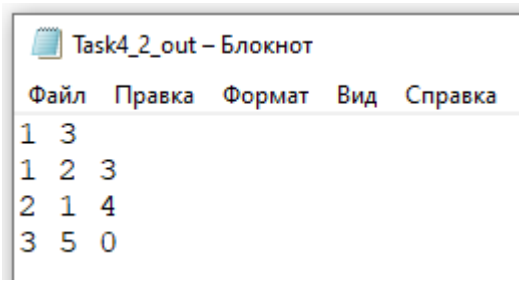


Рисунок 4.3 – Пример файла Task4_2_out.txt.

После успешной обработки вывести на экран содержимое файлов Task4_in.txt (исходные данные), Task4_1_out.txt (симметричные матрицы), Task4_2_out.txt (несимметричные).

При отсутствии одной из групп (например, все матрицы оказались несимметричными) соответствующий файл создаётся с $k_s = 0$ (или $k_{ns} = 0$) и без строк с матрицами. Это не ошибка. Если входные данные некорректны, нужно вывести сообщение об ошибке (например, «Ошибка формата данных») и прервать выполнение без записи выходных файлов.

Следует обратить внимание, что все входные данные (файл Task4_in.txt) формируются вручную в соответствии с указанным форматом. Чтобы при отладке программы обеспечить правильную обработку ошибок, некоторые данные могут быть некорректными (нечисловые символы, неверное количество строк или столбцов, недопустимые размеры и т. д.).

Задание 5 (35 баллов)

Написать программу, которая определяет ситуацию МАТ черному королю на шахматной доске. На шахматной доске может находится любое количество черных и белых фигур (пешка, слон, конь, ладья или ферзь), черный король находится обязательно.

Определение:

МАТ — ситуация в шахматах, когда король находится под шахом, и игрок не может сделать ни одного хода, чтобы его избежать (пример мата на рисунке 5.1). Таким образом, при мате одновременно:

король находится под шахом;

у короля нет возможности уйти от шаха - все поля вокруг него заняты своими фигурами или находятся под ударом фигур противника.

у игрока нет возможности закрыться от шаха другой фигурой.

нет возможности взять фигуру, объявившую шах.

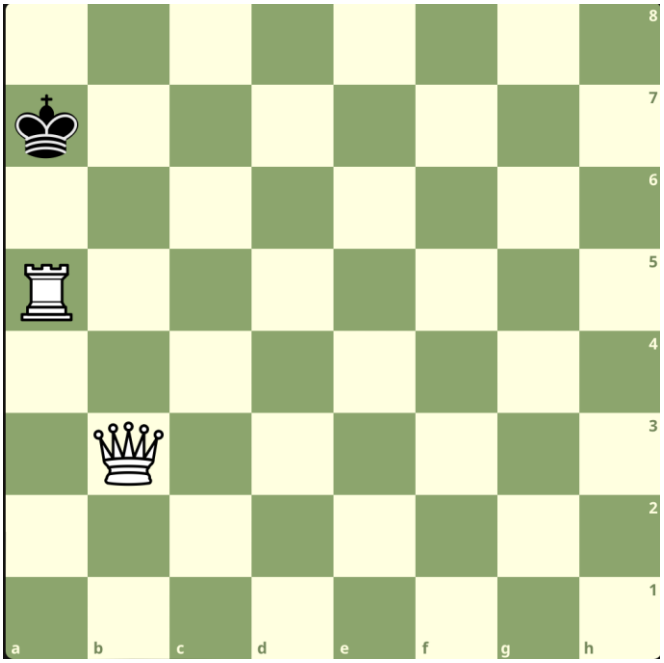


Рисунок 5.1 – Пример ситуации МАТ

Обозначения:

Игровое поле для игры в шахматы представляет собой доску, разделенную на 64 квадратные клетки по 8 клеток по горизонтали и 8 клеток по вертикали. Каждая вертикаль обозначается латинской буквой от a до h, а каждая горизонталь - арабской цифрой от 1 до 8. Таким образом каждая клетка имеет свой уникальный адрес, например b2 или e4 (представлена на рисунке 5.2.).

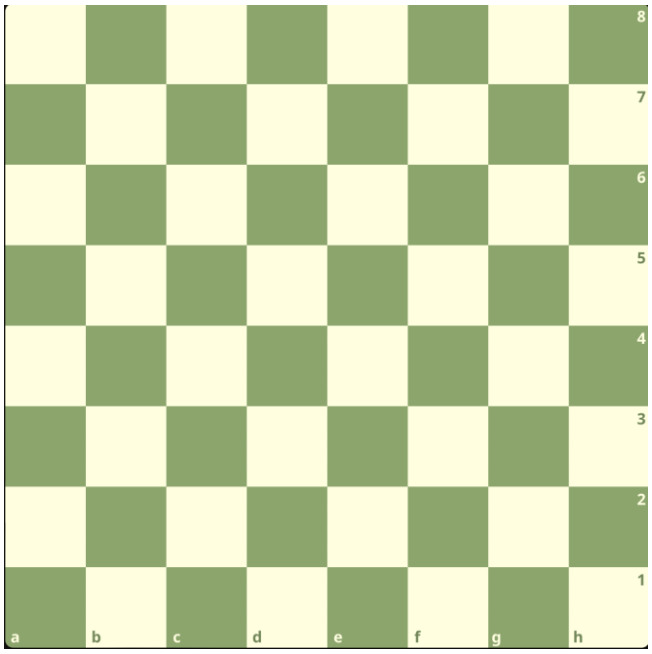


Рисунок 5.2 – Шахматная доска

Шахматная фигура может находится на одной определенной клетке доски. На одной клетке может находится только одна фигура. Для записи положения фигур будут использоваться стандартное обозначение - имя фигуры и имя клетки, на которой располагаются фигуры. Фигуры обозначаются так: N - конь, B - слон, R - ладья, Q - ферзь, K - король. Пешка никак не обозначается. Так, например, запись Nb1 означает, что конь

находится на клетке b1, запись e8 означает, что пешка находится на клетке e8. Пробел между именем фигуры и именем клетки не ставится.

Правила игры:

Пешки могут двигаться только на одну клетку по вертикали. Белые пешки двигаются “вверх” - в направлении увеличения числа в имени клетки, а черные - “вниз” - в направлении уменьшения. Назад пешки ходить не могут. Исключение - белые пешки, находящиеся на 2 горизонтали, и черные - на 7 горизонтали. Эти пешки могут ходить как на одну клетку вперед, так и на две.

Конь движется на две клетки в одном из четырех направлений и на одну клетку в одном их двух, направлений, перпендикулярном первому:

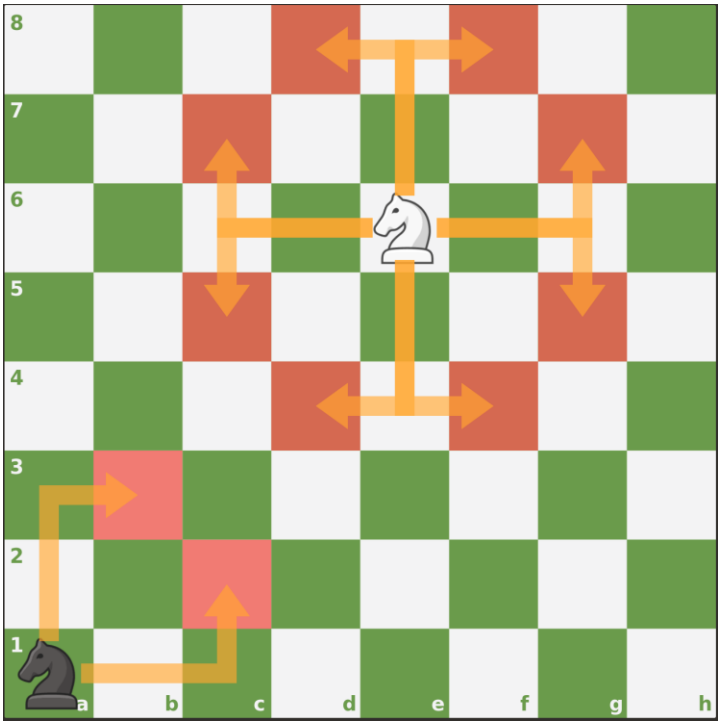


Рисунок 5.3 – Ход фигуры Конь

На рисунке 5.3 красным отмечены клетки, на которые может переместиться соответствующий конь. Конь, расположенный в середине доски может переместиться на любую из восьми клеток. Конь. расположенный в углу может переместиться на любую из двух клеток потому, что фигуры не могут выходить за пределы доски.

Ладья движется по горизонтали и вертикали на любое доступное количество клеток.

Слон движется по горизонтали на любое доступное количество клеток.

Фигуры не могут двигаться за пределы доски. Фигуры за исключением коня не могут “перепрыгивать” через другие фигуры, в том числе короля.

Шах - это ситуация, когда король находится на клетке под боем фигуры или пешки противоположного цвета. Клетка считается под боем фигуры тогда, когда эта фигура может следующим ходом переместиться со своего текущего положения на эту клетку. Исключение - пешки. Под боем пешки находятся клетки по диагонали:

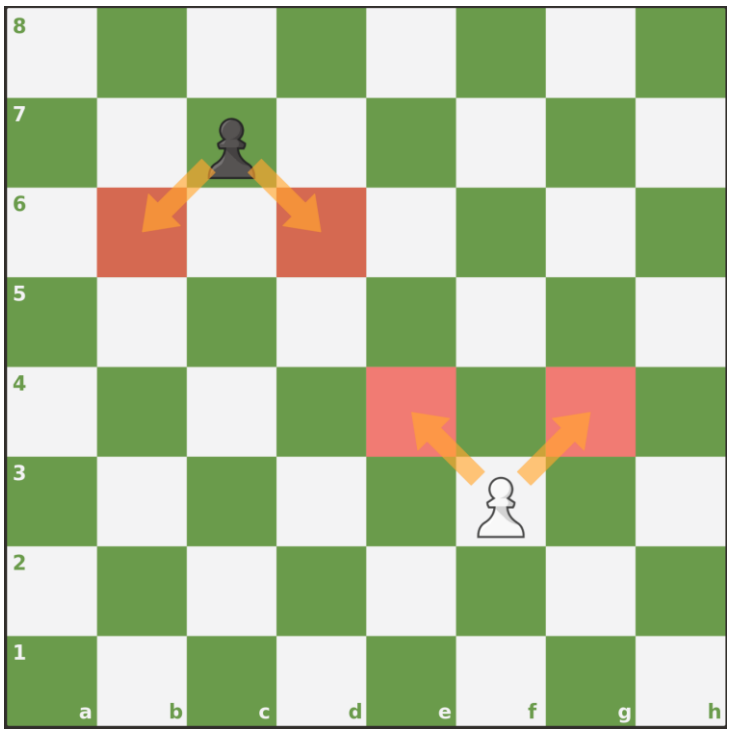


Рисунок 5.4 – Клетки под боем фигуры пешка

На рисунке 5.4 красным обозначены клетки, находящиеся под боем ближайших пешек. Обратите внимание, что направление боя для пешки зависит от ее цвета и, как следствие, направления движения.

Формат входного файла:

Входной файл состоит из двух строк:

- в первой строке обязательно указывается положения черного короля и любое количество черных фигур, разделенных пробелом (допускается указать положение только черного короля);
- во второй строчке указываются положения белых фигур, количество фигур равно одному и более, разделённые пробелом

Пример файла:

a7 Ka8 Bb6

Nd7 Qd5

соответствует такому положению на доске (рисунок 5.5):

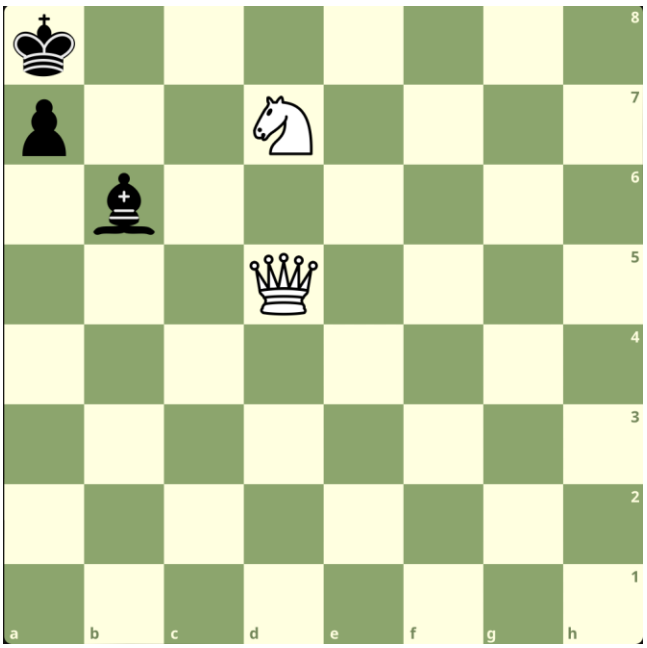


Рисунок 5.5

А такой файл:
Kd6 Bc5
Ba3 Re3
соответствует такому положению на доске (рисунок 5.6):

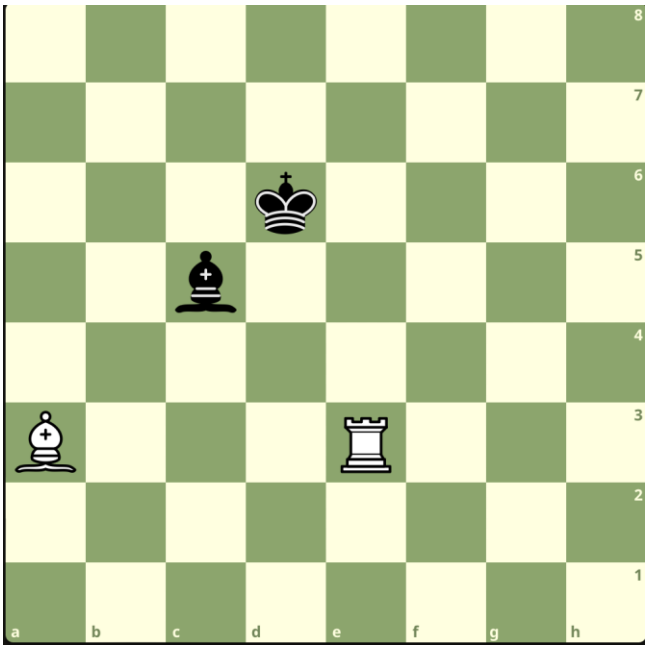


Рисунок 5.6 -

Формат выходного файла:

Если ситуация на шахматной доске определена как МАТ, то программа должна записать в выходной файл 1, иначе 0.

Первый пример:

Входной файл:

a7 Ka8 Bb6
Nd7 Qd5

Выходной файл должен содержать 1.

Второй пример:
Входной файл:

Kd6 Bc5
Ba3 Re3

Выходной файл должен содержать 0.