ФГОБУ ВО «Финансовый университет при Правительстве Российской Федерации»
ВСЕРОССИЙСКАЯ ОЛИМПИАДА ПО ИНФОРМАТИКЕ
«МИССИЯ ВЫПОЛНИМА. ТВОЕ ПРИЗВАНИЕ – ФИНАНСИСТ!»
ОЧНЫЙ ЭТАП, 2024 год

Продолжительность олимпиады – 240 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**)**

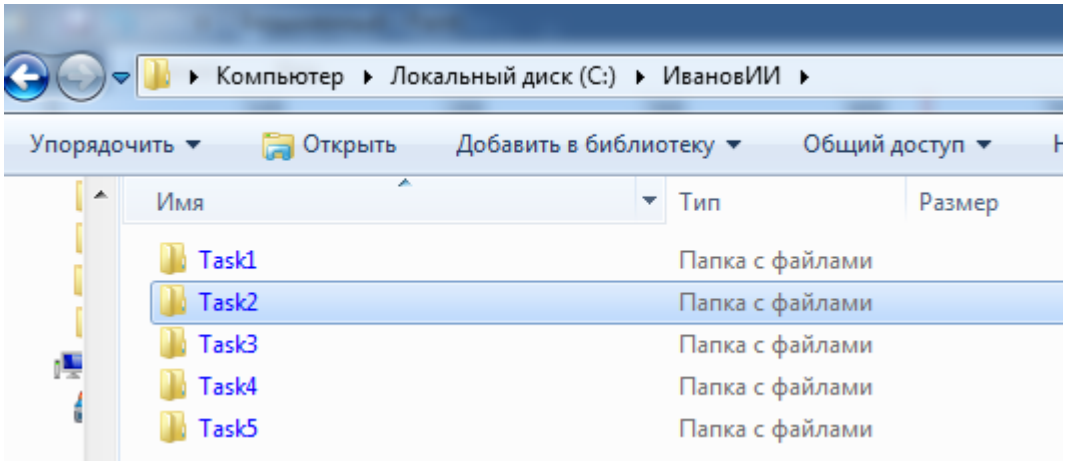


Рисунок П.1 – Структура каталога участника олимпиады

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)

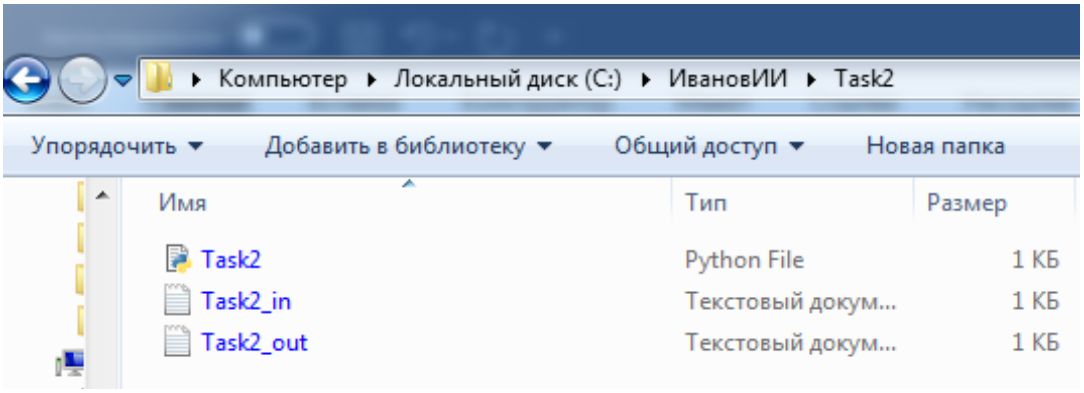


Рисунок П.2 – Размещение файлов в рамках папки задачи

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).

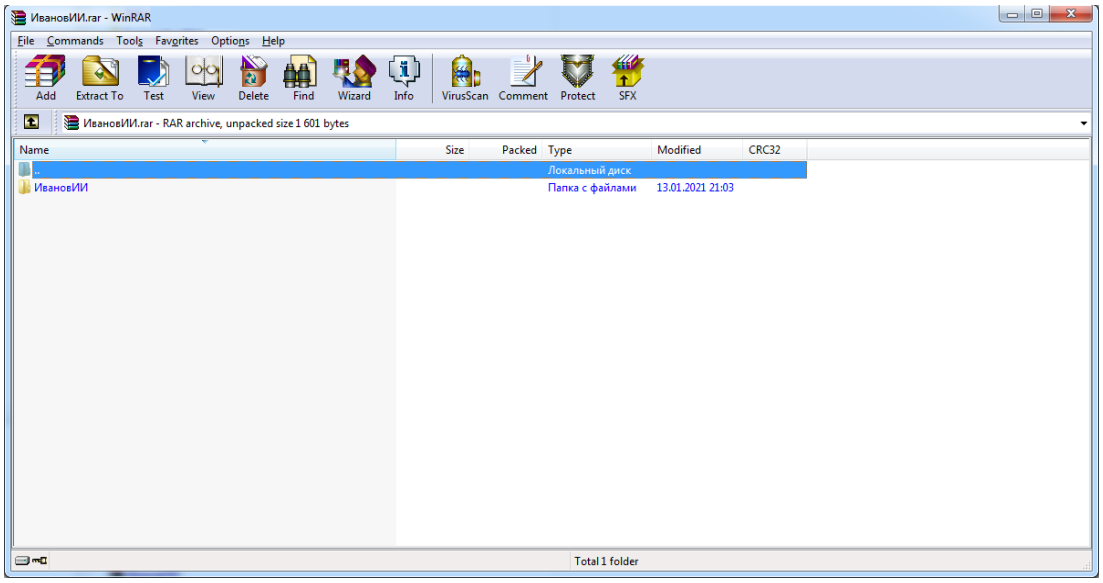


Рисунок П.3 – Пример первого уровня содержимого архива

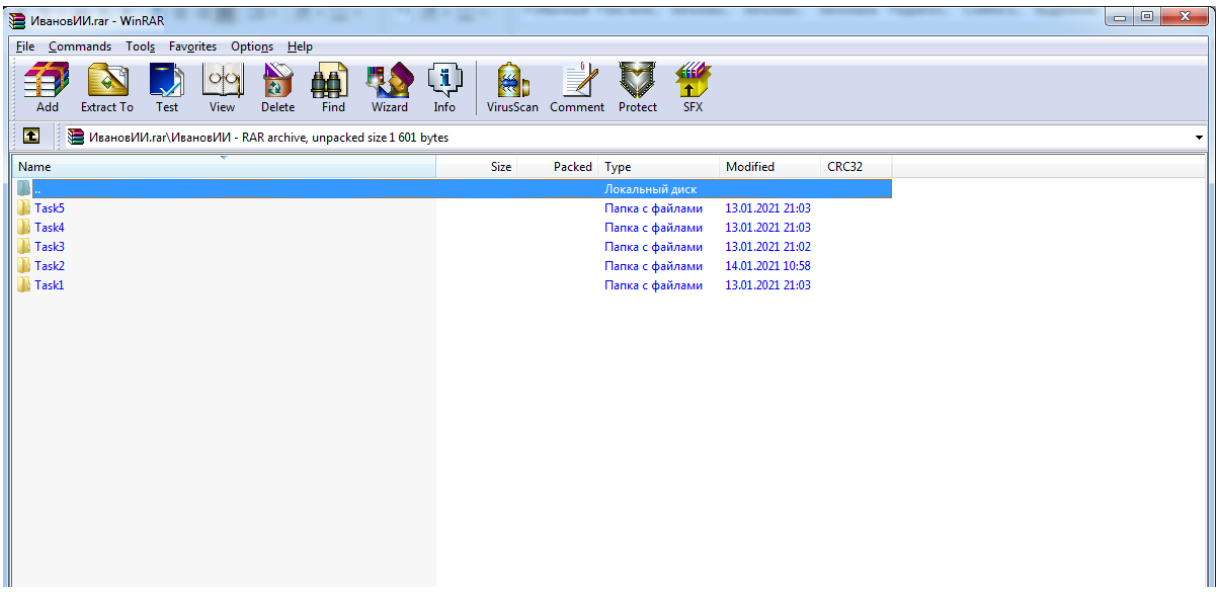


Рисунок П.3 – Пример второго уровня содержимого архива

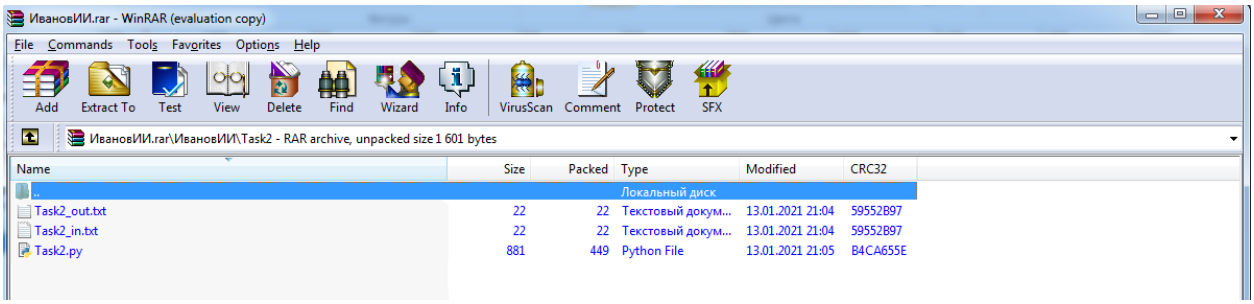


Рисунок П.3 – Пример третьего уровня содержимого архива

Члены комиссии, при проверке заданий, используют операционные системы семейства Windows (версии 7-10). 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.

$$y = \frac{\sqrt{(4 - x)}}{x^2 - 25} + log_4(x + 3)$$

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 = (A or B or C) and (C and D) or (not A ≡C) ←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;
- ≡ – эквивалентность.

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).

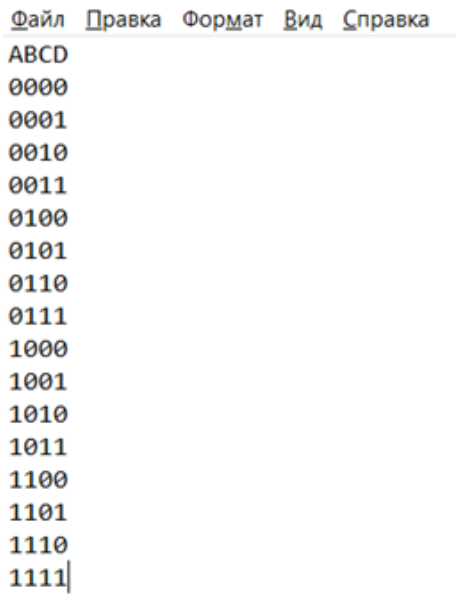
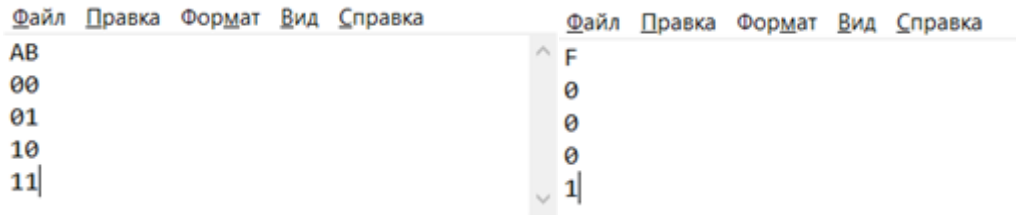


Рисунок 2.2 – Файл Task2_in.txt

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 решили использовать шифр Плейфера. Длина сообщений при этом никогда не превышает 25 символов, включая пробелы.

Шифр Плейфера использует матрицу 5x5, содержащую ключевое слово или фразу. Для создания матрицы и использования шифра достаточно запомнить ключевое слово и четыре простых правила. Чтобы составить ключевую матрицу, в первую очередь нужно заполнить пустые ячейки матрицы буквами ключевого слова (не записывая повторяющиеся символы), потом заполнить оставшиеся ячейки матрицы символами алфавита, не встречающимися в ключевом слове, по порядку (в английских текстах обычно опускается символ «Q», чтобы уменьшить алфавит, в других версиях «I» и «J» объединяются в одну ячейку). Ключевое слово может быть записано в верхней строке матрицы слева направо, либо по спирали из левого верхнего угла к центру. Ключевое слово, дополненное алфавитом, составляет матрицу 5x5 и является ключом шифра.

Для того чтобы зашифровать сообщение, необходимо разбить его на биграммы (группы из двух символов), например «Hello World» становится «HE LL OW OR LD», и отыскать эти биграммы в таблице. Два символа биграммы соответствуют углам прямоугольника в ключевой матрице. Определяем положения углов этого прямоугольника

относительно друг друга. Затем, руководствуясь следующими 4 правилами, зашифровываем пары символов исходного текста:

1. Если два символа биграммы совпадают (или если остался один символ), добавляем после первого символа «X», зашифровываем новую пару символов и продолжаем.

2. Если символы биграммы исходного текста встречаются в одной строке, то эти символы замещаются на символы, расположенные в ближайших столбцах справа от соответствующих символов. Если символ является последним в строке, то он заменяется на первый символ этой же строки.

3. Если символы биграммы исходного текста встречаются в одном столбце, то они преобразуются в символы того же столбца, находящиеся непосредственно под ними. Если символ является нижним в столбце, то он заменяется на первый символ этого же столбца.

4. Если символы биграммы исходного текста находятся в разных столбцах и разных строках, то они заменяются на символы, находящиеся в тех же строках, но соответствующие другим углам прямоугольника.

Для расшифровки необходимо использовать инверсию этих четырёх правил, откидывая символы «X», если они не несут смысла в исходном сообщении.

Пример работы данного алгоритма шифрования, следующий.

Фраза, которую требуется зашифровать: FOR EXAMPLE.

Ключ для шифрования: TABLE.

Тогда получаем матрицу, представленную на рисунке 3.1.

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

Рисунок 3.1

Зашифрованный текст: HMUBWBIRET.

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. 3.2):

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

Рисунок 3.2

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 3.3.

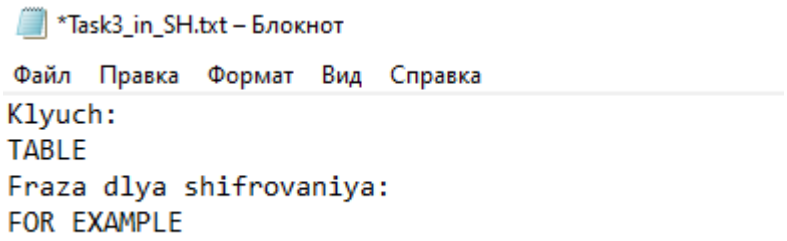


Рисунок 3.3

В качестве выходного файла формируется файл Task3_out_SH.txt структура которого представлена на рисунке 3.4.

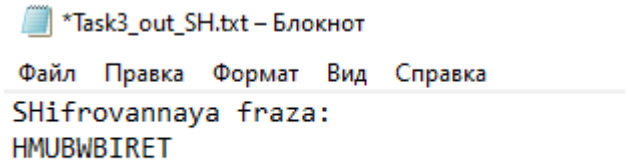


Рисунок 3.4

При дешифровании в качестве входного файла программа должна использовать файл Task3_in_DSH.txt структура которого представлена на рисунке 3.5.

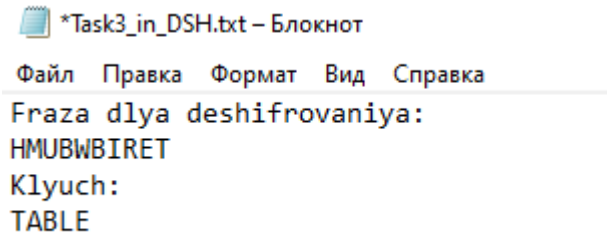


Рисунок 3.5

В качестве выходного файла при дешифровании формируется файл Task3_out_DSH.txt структура которого представлена на рисунке 3.6.

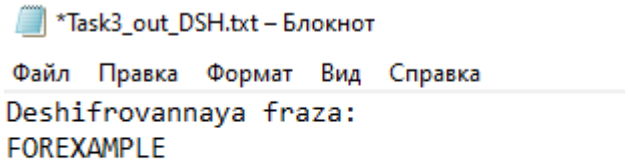


Рисунок 3.6

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 20 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)

Зерноуборочный комбайн ездит по полю по часовой стрелке по спирали и собирает в пшеницу. Размер поля имеет длину N и ширину M, форма поля может быть прямоугольной или квадратной. Начинает собирать пшеницу комбайн в **точке с координатами (i, j) равными (1, 1),** т.е. i=1 и j=1. Отсчет времени всегда начинается с 1. Через равные промежутки времени он переезжает на новый участок поля. Например, если поле имеет длину N=5 и ширину M=4, комбайн будет перемещаться по пути, как показано в таблице:

	i=1	i=2	i=3	i=4	i=5
j=1	1	14	13	12	11
j=2	2	15	20	19	10
j=3	3	16	17	18	9
J=4	4	5	6	7	8

При построении пути комбайна нам известно, что на участке с координатами (4,3) он будет через 18 промежутков времени.

К сожалению, на сельхоз предприятии для отчетности необходимо предоставить четкие размеры поля и движение комбайна по нему. Заранее известна только ширина поля, но не известна ее длина. А так же известно, в какой момент времени комбайн будет на участке с координатами (i,j). Предусмотреть что поле имеет длину минимум равную 3.

Входные данные: Ширина поля равна 5, комбайн будет на участке с координатами i=4 и j=3 через 18 промежутков времени

В качестве входного файла программа должна использовать файл Task4_in.txt структура которого представлена на рисунке 4.1:

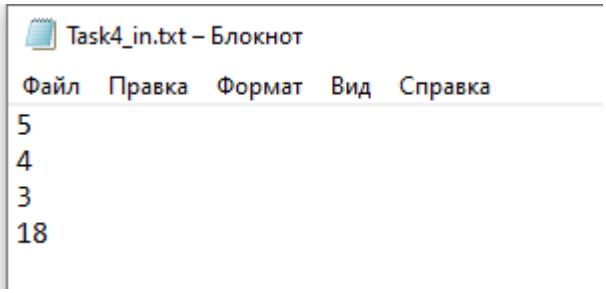


Рисунок 4.1

Пример выходного файла Task4_out.txt представлен на рисунке 4.2:

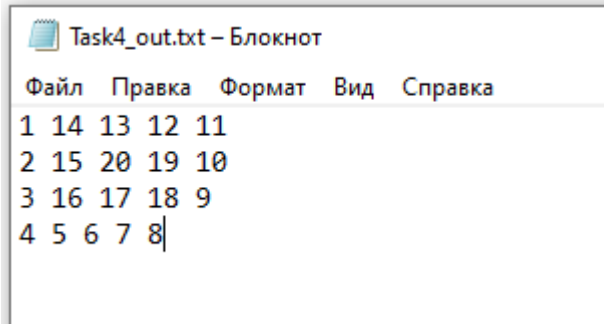


Рисунок 4.2

Предусмотреть вывод сообщения об ошибке, если:

1. Пользователь не ввел 4 натуральных числа.
2. Пользователь ввел ширину меньше 3 или не натуральное число.
3. Пользователь ввел не корректные координаты участка (не натуральные положительные числа).
4. Пользователь ввел буквы вместо цифр.
5. Вывод сообщения об ошибке, если данные введены не корректно и произошло зацикливание программы.

Задание 5 (35 баллов)

На шахматной доске находится черный король и белая фигура (слон, конь, ладья, ферзь или король). Требуется написать программу, которая определяет, может ли белая фигура за один ход дать шах черному королю при условии, что сейчас ход белых, и определить ход белой фигуры, объявляющий шах. При этом не должны учитываться ходы, при которых белая фигура сама становится под бой черного короля.

Описание

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 5.1.).

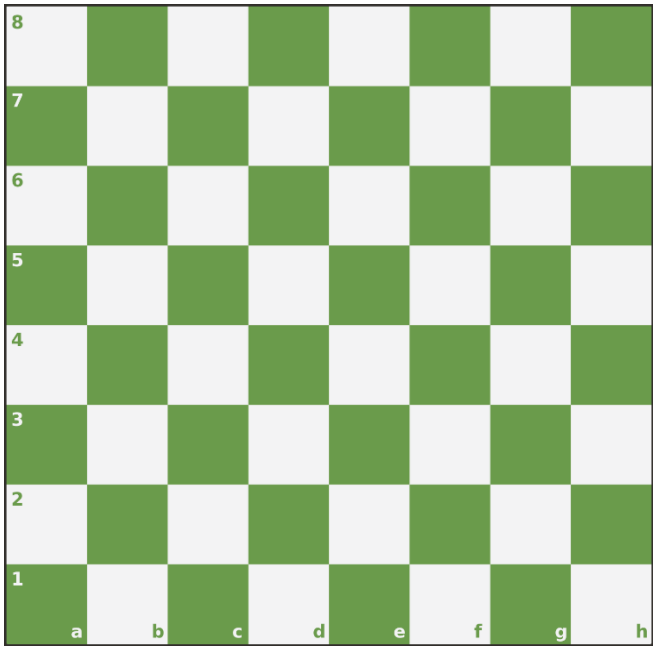


Рисунок 5.1

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:

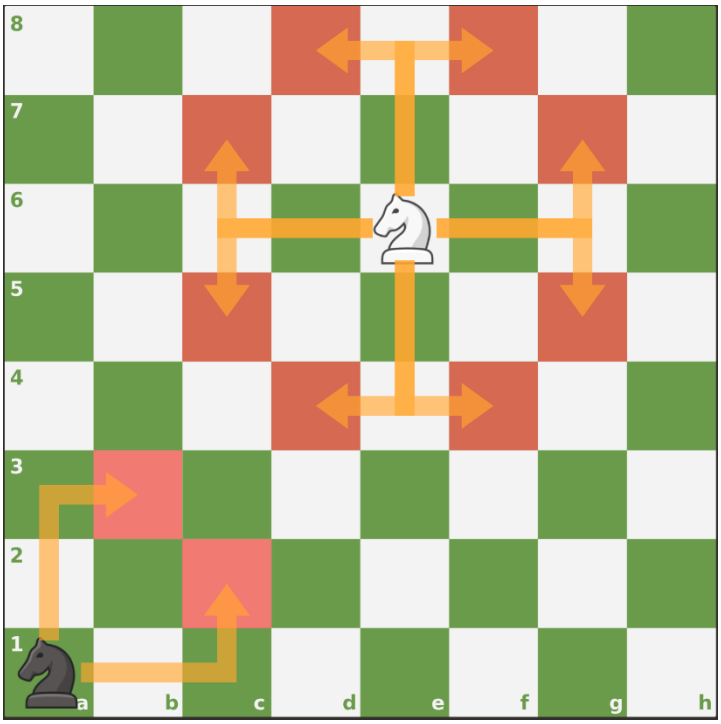


Рисунок 5.2

На рисунке 5.2 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 диагонали на любое доступное количество клеток.

Ферзь может ходить на любое количество клеток либо по горизонтали, либо по вертикали, либо по диагонали, то есть или как ладья или как слон.

Король может ходить на одну клетку в любом направлении. То есть за один ход король может переместиться на любое из восьми соседних полей.

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 по направлению движения пешки:

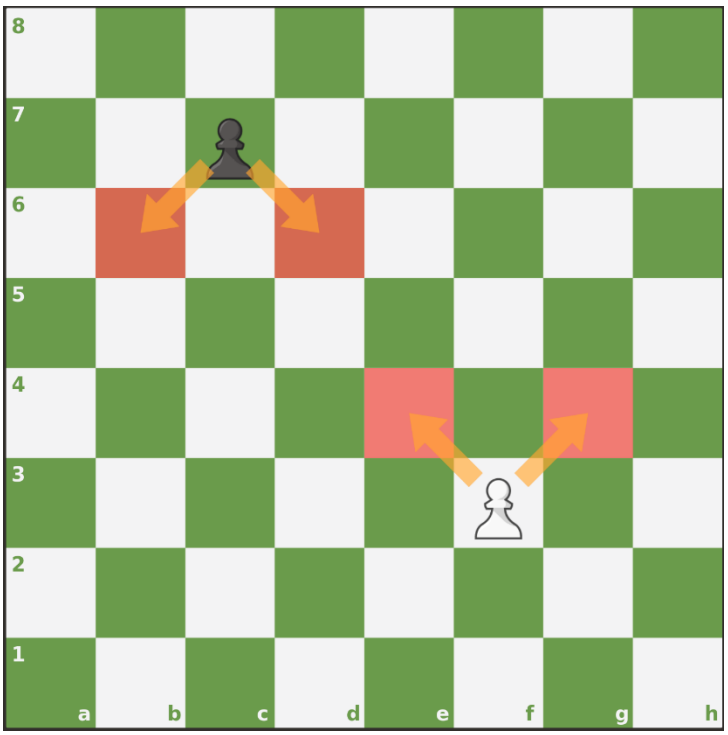


Рисунок 5.3

На рисунке 5.3 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:

Во входном файле указаны положения белой фигуры и черного короля в одну строчку через пробел. Пример:

e4 Kf8

соответствует такому положению на доске (рисунок 5.4):

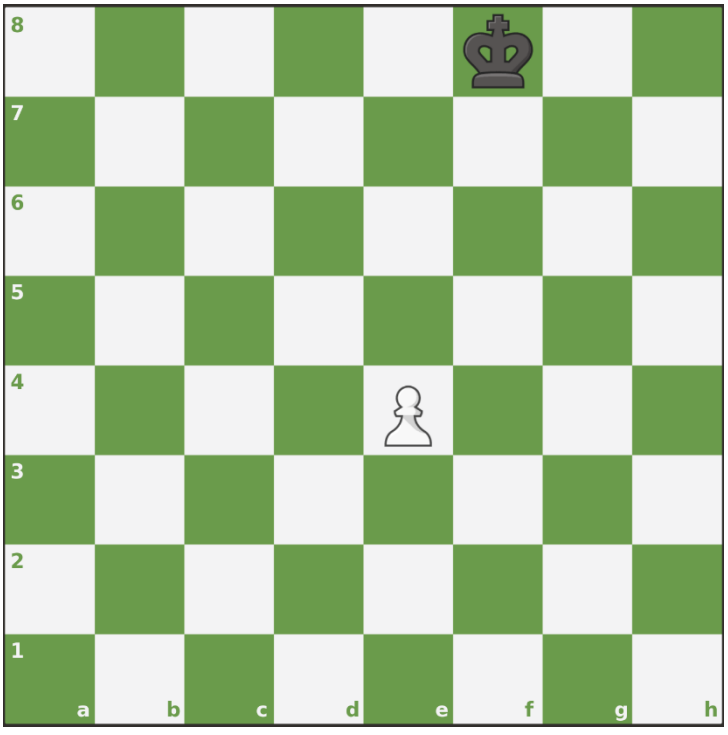


Рисунок 5.4

А такой файл:

Qg7 Kc3

соответствует такому положению на доске (рисунок 5.5):

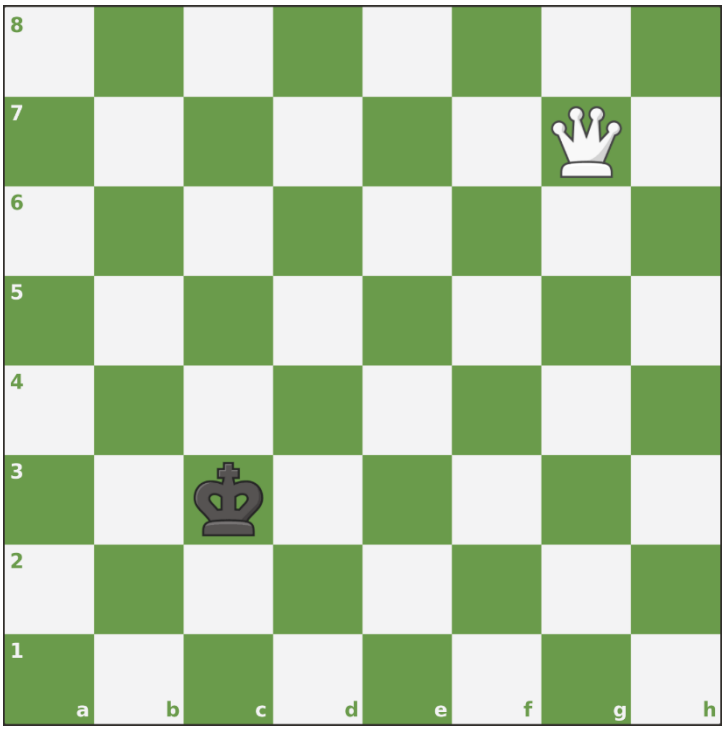


Рисунок 5.5

Формат выходного файла:

Если шах возможен, то программа должна записать в выходном файле соответствующих ход, то есть положение фигуры, на которое она должна переместиться для объявления шаха. Если таких ходов несколько, то программа должна вывести первый по алфавиту. Если черный король уже находится под шахом в исходном положении (даже если сама белая фигура находится под боем), программа должна вывести 0.

Если шах невозможен, то выходной файл должен быть пустым.

Пример: входной файл - “Be6 Ke2”. Выходной файл должен содержать “Bc4”.

ФГОБУ ВО «Финансовый университет при Правительстве Российской Федерации»
ВСЕРОССИЙСКАЯ ОЛИМПИАДА ПО ИНФОРМАТИКЕ
«МИССИЯ ВЫПОЛНИМА. ТВОЕ ПРИЗВАНИЕ – ФИНАНСИСТ!»
ОЧНЫЙ ЭТАП, 2024 год

Продолжительность олимпиады – 240 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**)**

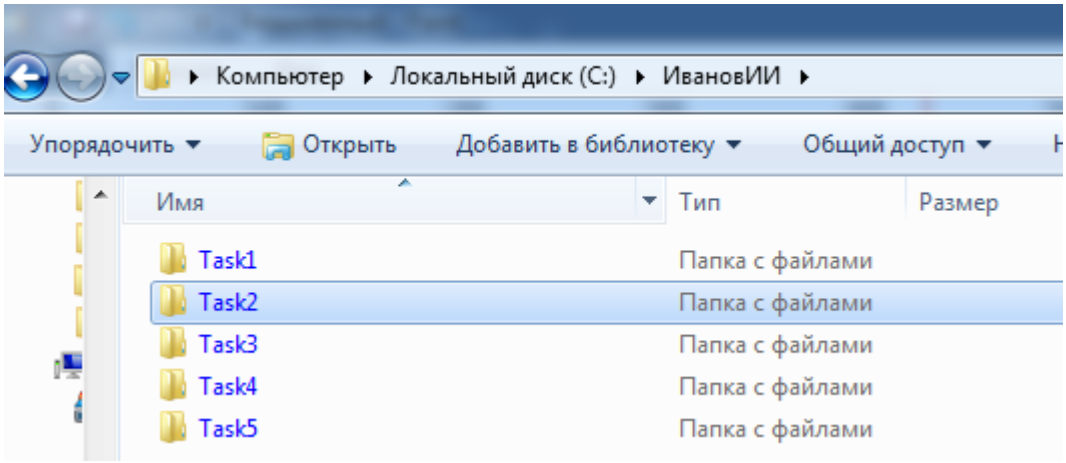


Рисунок П.1 – Структура каталога участника олимпиады

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)

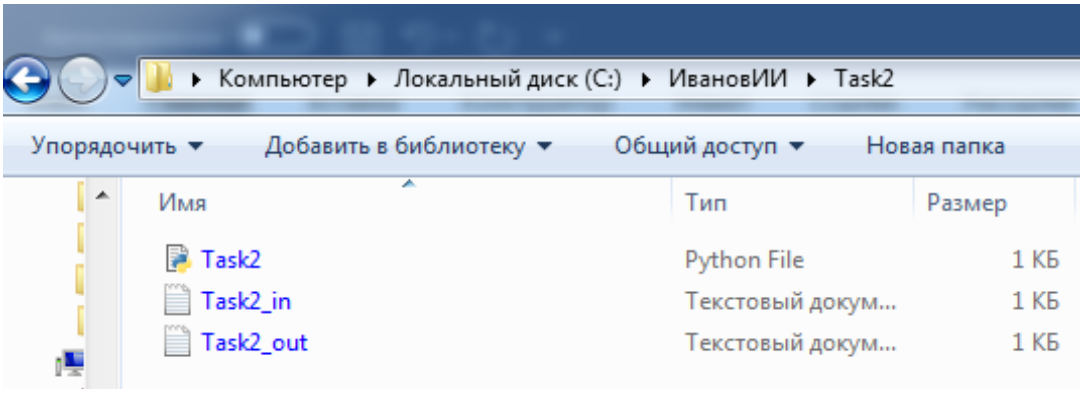


Рисунок П.2 – Размещение файлов в рамках папки задачи

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).

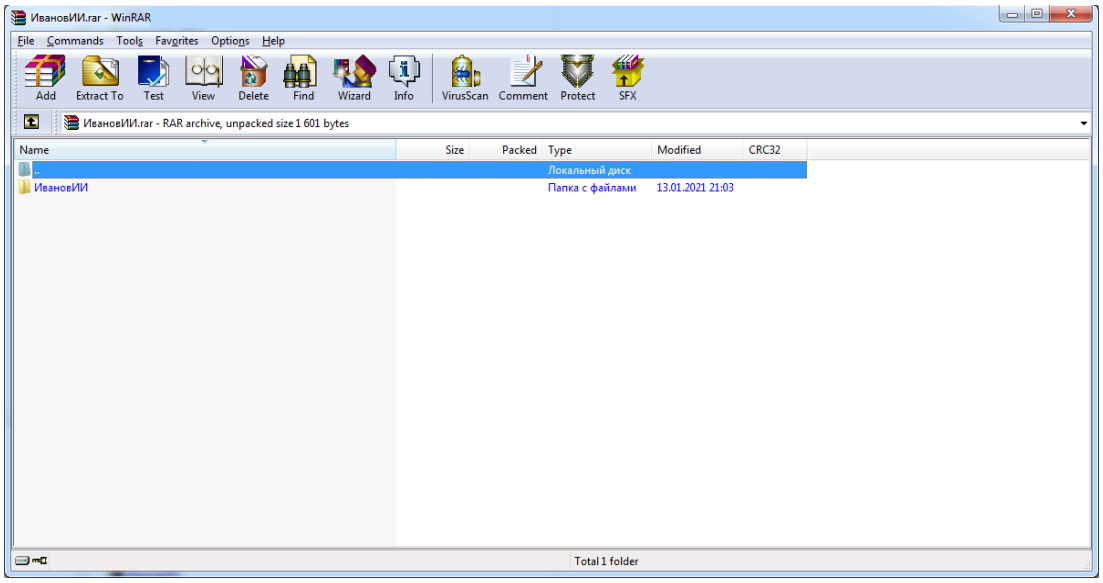


Рисунок П.3 – Пример первого уровня содержимого архива

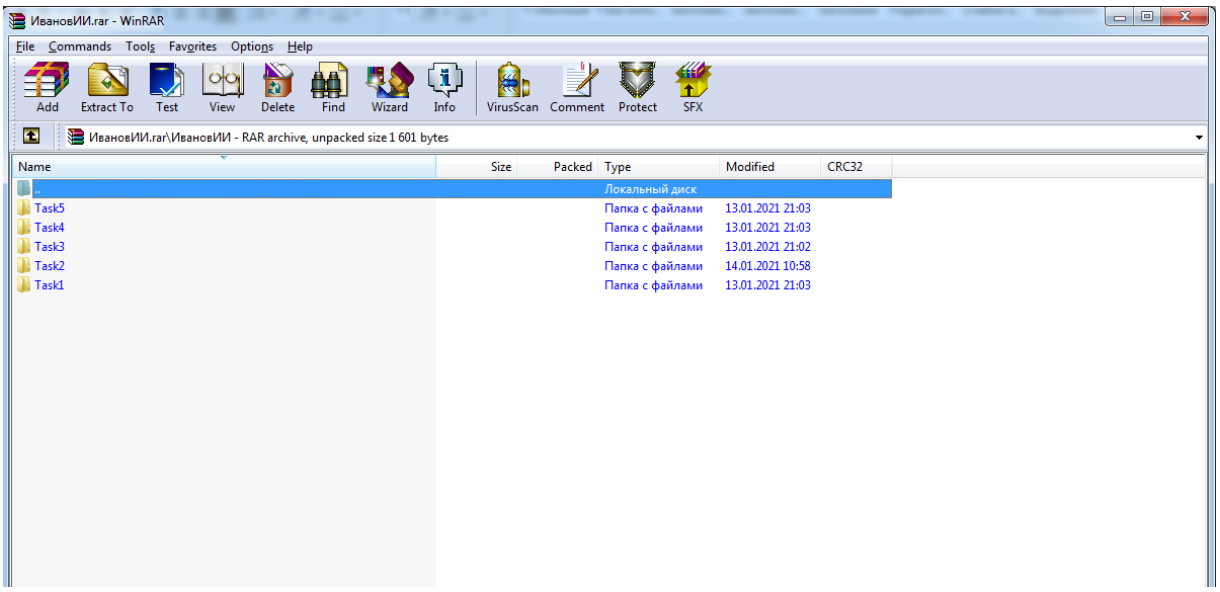


Рисунок П.3 – Пример второго уровня содержимого архива

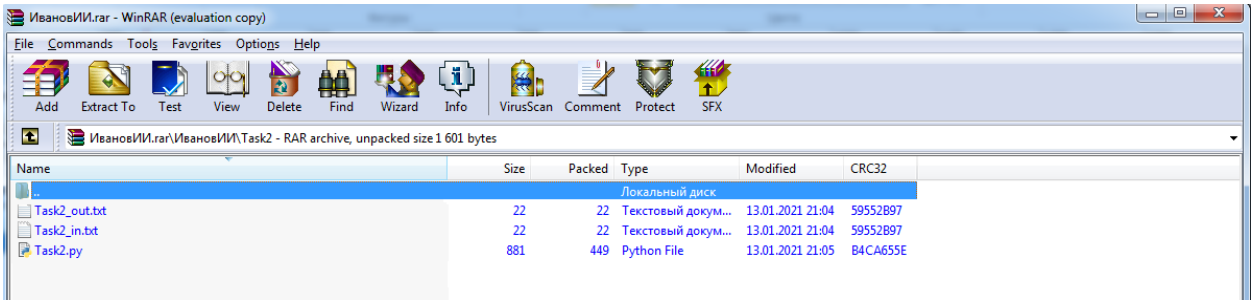


Рисунок П.3 – Пример третьего уровня содержимого архива

Члены комиссии, при проверке заданий, используют операционные системы семейства Windows (версии 7-10). 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 №2

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.

$$y = \frac{\sqrt{(16 - x^2)}}{8 - x^3} + log_5(2x + 3)$$

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 = (A or B and C) or (C and D) or (not B ≡ not C) ← D

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;
- ≡ – эквивалентность.

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).

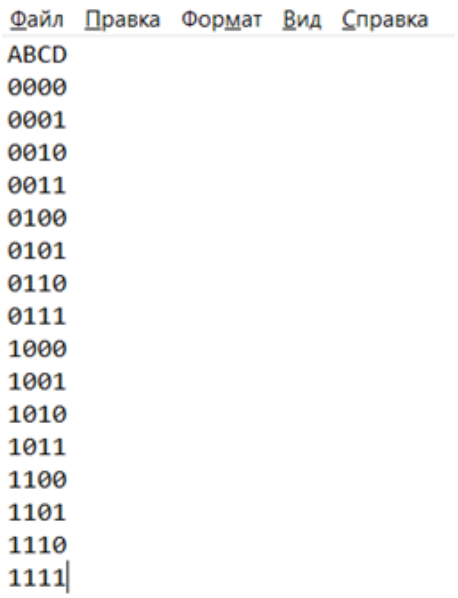
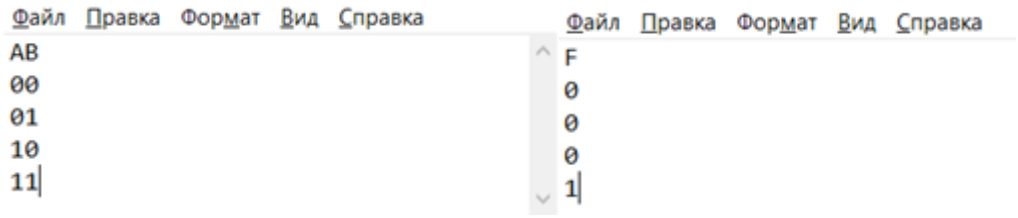


Рисунок 2.2 – Файл Task2_in.txt

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 филиалами Финуниверситета решили использовать шифр Уитстона. Длина сообщений при этом никогда не превышает 20 символов, включая пробелы.

Для шифрования методом Уитстона используются две матрицы 5x5, которые находятся либо одна под другой в (вертикальном варианте), либо друг напротив друга (в горизонтальном). Каждая из матриц в первую очередь заполняется соответствующим ключевым словом, не записывая повторяющиеся буквы, затем в оставшиеся ячейки матрицы записываются по порядку символы алфавита, которые не были ранее использованы (обычно буква «Q» опускаются, чтобы уменьшить алфавит, либо «I» и «J» объединяются в одной клетке). Ключевое слово может быть записано двумя способами: либо в верхней строке матрицы слева направо, либо по спирали из левого верхнего угла к центру. Таким образом полностью заполняется матрица 5x5. Стоит отметить, что ключ шифра Уитстона, как правило, состоит из двух слов (по одному слову на каждую матрицу).

Чтобы зашифровать сообщение, необходимо выполнить следующие действия:

Разбиваем сообщение на биграммы (группы из двух символов).

В вертикальном способе шифрования первый символ биграммы находим в верхней матрице, второй — в нижней.

В горизонтальном способе шифрования первый символ находим в левой матрице, второй — в правой.

Определяем положения углов получившегося прямоугольника относительно друг друга. В случае, если буквы исходной биграммы сообщения находятся в одной строке (в горизонтальном шифровании), то первую букву шифрованной биграммы берут из левой матрицы в том по счету столбце, в каком находится вторая буква исходной биграммы. Вторая же буква шифрованной биграммы берется из второй матрицы в столбце, по счету которого находится первая буква исходной биграммы сообщения. Аналогичным образом поступаем в случае вертикального шифрования.

В качестве примера, ниже приведен вертикальный шифр Уитстона с ключевыми словами «example» и «keyword».

Для шифрования и расшифрования используются матрицы, представленные на рисунке 3.1.

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

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

Рисунок 3.1

Допустим, необходимо зашифровать открытый текст hello world. Биграммы этого сообщения будут заменяться следующим образом:

1. Биграмма he имеет уникальный случай, она расположена в одном столбце, заменяем её на XG.

2. Биграмма ll тоже имеет уникальный случай, она расположена в первом столбце, заменяем её на NR.

3. Биграмма ow образует прямоугольник, заменяем её на SE.

4. Биграмма or образует прямоугольник, заменяем её на ND.

5. Биграмма ld образует прямоугольник, заменяем её на BR.

Таким образом, получаем шифрованное сообщение: XGNRSENDBR

Метод для расшифровки идентичен методу зашифрования. Для этого при шифровании вместо открытого текста используется ранее полученный шифротекст, в результате чего получается исходный открытый 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. 3.2):

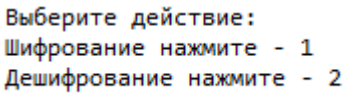


Рисунок 3.2

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 3.3.

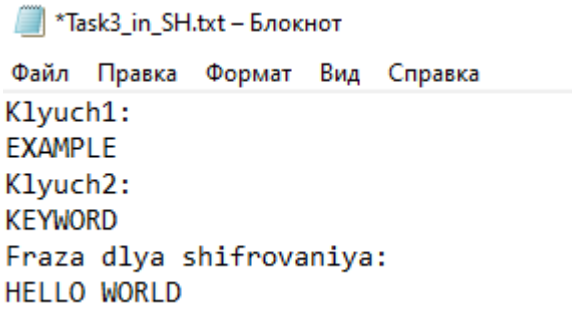


Рисунок 3.3

В качестве выходного файла формируется файл Task3_out_SH.txt структура которого представлена на рисунке 3.4.

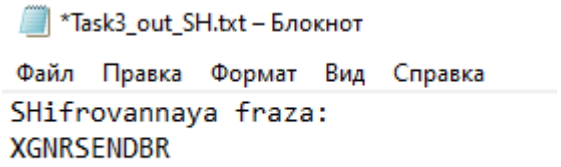


Рисунок 3.4

При дешифровании в качестве входного файла программа должна использовать файл Task3_in_DSH.txt структура которого представлена на рисунке 3.5.

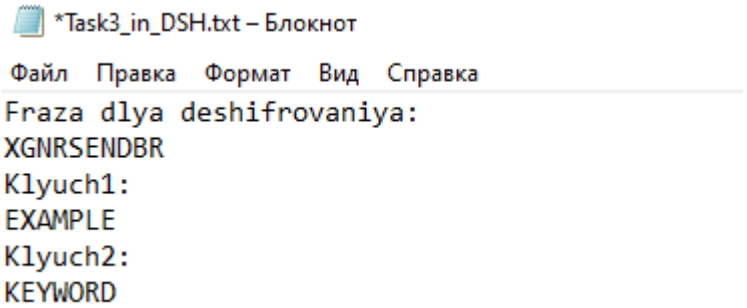


Рисунок 3.5

В качестве выходного файла при дешифровании формируется файл Task3_out_DSH.txt структура которого представлена на рисунке 3.6.

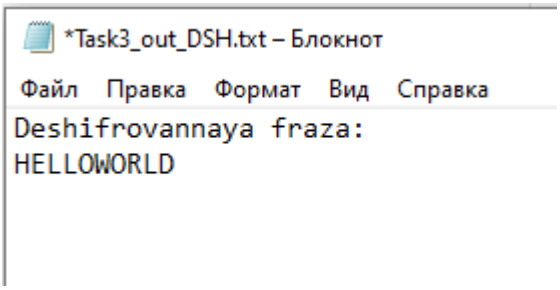


Рисунок 3.6

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 20 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)

Экскаватор разрабатывает карьер. Он движется по карьеру по часовой стрелки. Карьер имеет имеет длину N и ширину M, форма карьера может быть квадратной или прямоугольной. Экскаватор начинает разрабатывать карьер в точке с координатами (i, j) равными (1, N), т.е. i=1 и j=N. Первый участок всегда разрабатывается на 1 метр в глубину, во второй на 2 метра и т.д.. Например, если карьер имеет длину N=5 и ширину M=4, экскаватор будет перемещаться по пути, как показано в таблице:

	i=1	i=2	i=3	i=4	i=5
j=1	11	12	13	14	1
j=2	10	19	20	15	2
j=3	9	18	17	16	3
j=4	8	7	6	5	4

При построении пути нам известно, что на участке с координатами (2, 3) глубина участка будет равна 18 метрам.

Когда определяли место для разработки они измерили только длину карьера, а в отчетности требуют предоставить рабочее задание с указанием схемы движения экскаватора и четкие размеры карьера. Но нам известно на сколько метров комбайн проработал участок с координатами (i,j). Предусмотреть что карьер имеет ширину минимум равную 3.

Входные данные: Длина карьера равна 4, комбайн проработал участок с координатами i=2 и j=3 на 18 метров.

В качестве входного файла программа должна использовать файл Task4_in.txt структура которого представлена на рисунке 4.1:

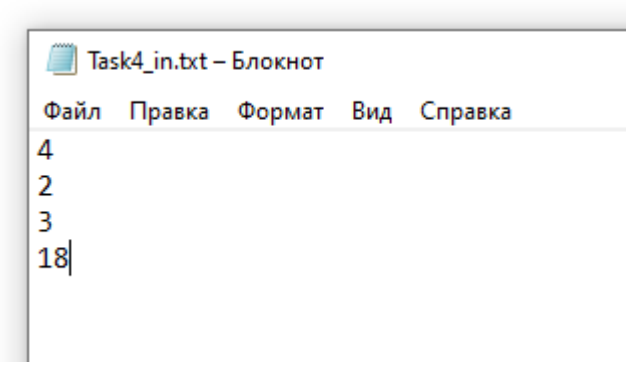


Рисунок 4.1

Пример выходного файла Task4_out.txt представлен на рисунке 4.2:

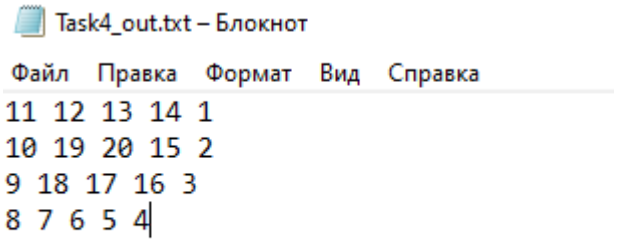


Рисунок 4.2

Предусмотреть вывод сообщения об ошибке, если:

1. Пользователь не ввел 4 натуральных числа.
2. Пользователь ввел длину меньше 3 или не натуральное число.
3. Пользователь ввел не корректные координаты участка (не натуральные положительные числа).
4. Пользователь ввел буквы вместо цифр.
5. Вывод сообщения об ошибке, если данные введены не корректно и произошло зацикливание программы.

Задание 5 (35 баллов)

На шахматной доске расположена одна белая фигура (пешка, слон, ладья, ферзь или король) и черный король. Требуется разработать программу, которая определяет, может ли данная фигура поставить шах черному королю и за сколько ходов при условии, что сейчас ход белых, а черный король не будет двигаться (то есть белые могут сделать несколько ходов подряд). При этом не должны учитываться ходы, при которых белая фигура сама становится под бой черного короля.

Описание

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 5.1.).

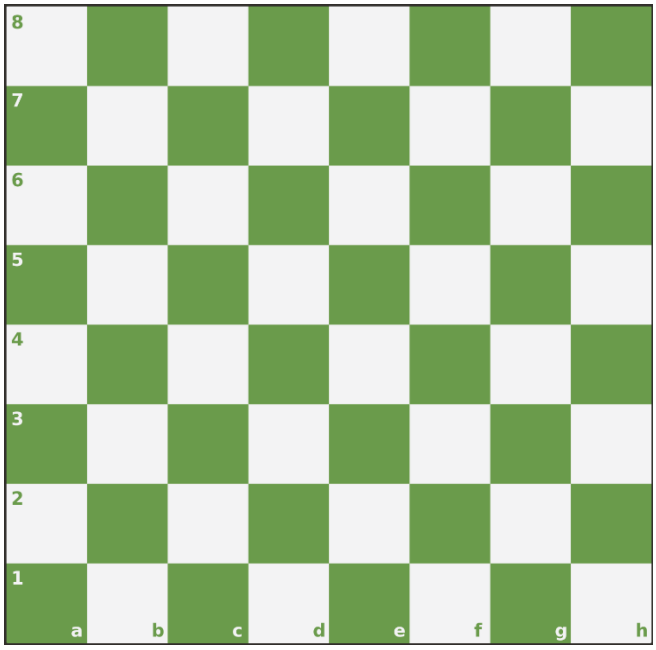


Рисунок 5.1

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:

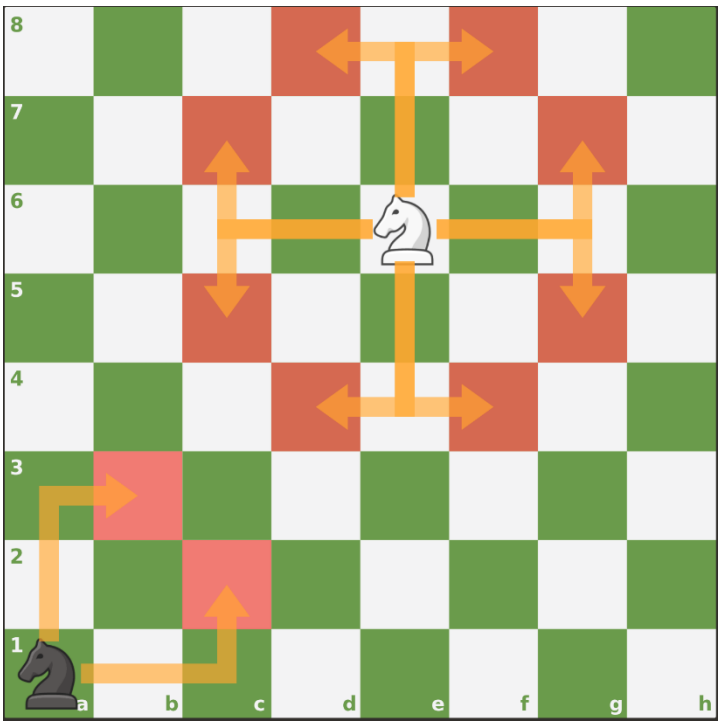


Рисунок 5.2

На рисунке 5.2 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 диагонали 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 по направлению движения пешки:

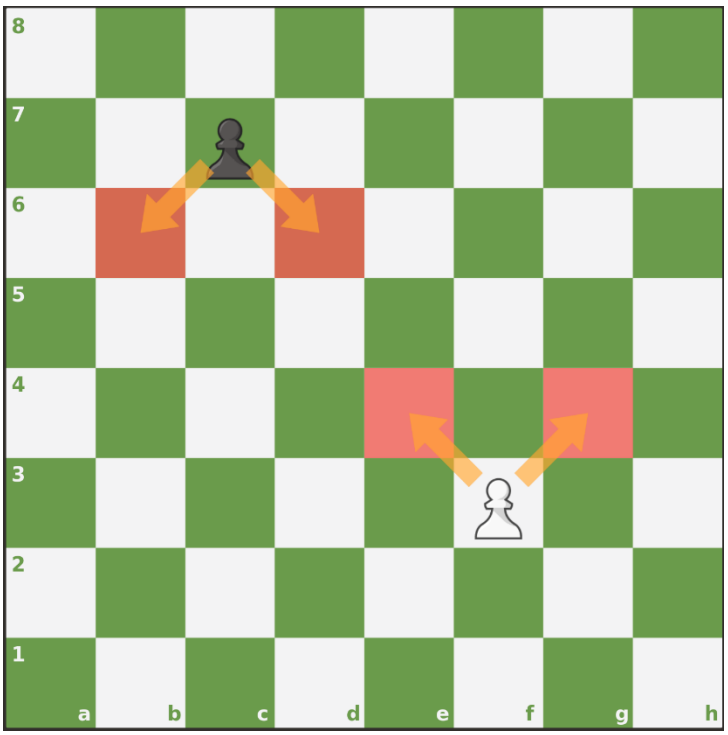


Рисунок 5.3

На рисунке 5.3 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:

Во входном файле указаны положения белой фигуры и черного короля в одну строчку через пробел. Пример:

e4 Kf8

соответствует такому положению на доске (рисунок 5.4):

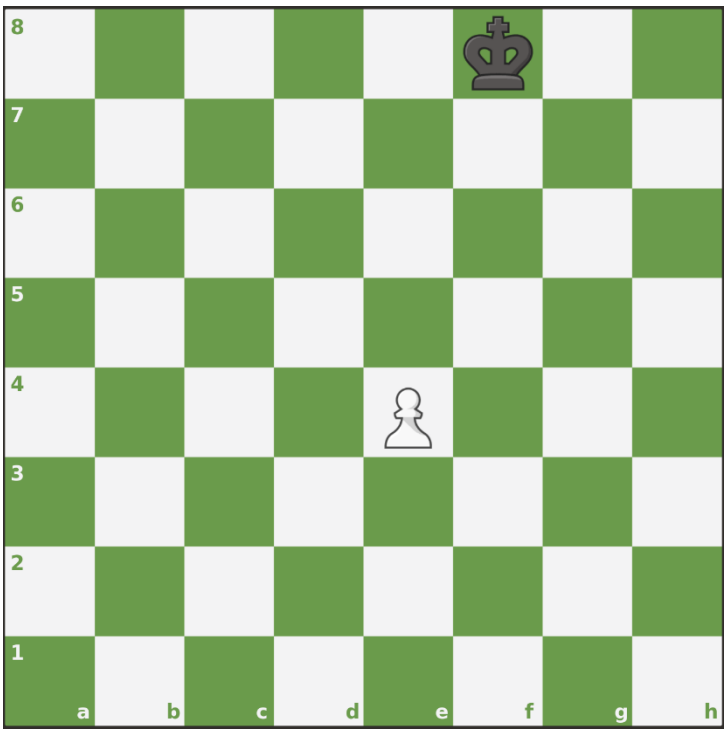


Рисунок 5.4

А такой файл:
Qg7 Kc3
соответствует такому положению на доске (рисунок 5.5):

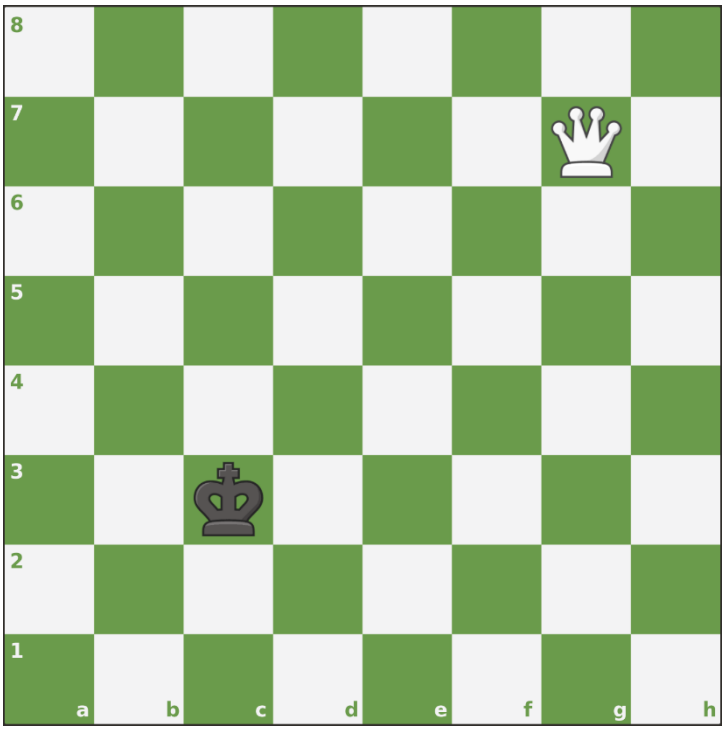


Рисунок 5.5

Формат выходного файла:

Если шах возможен, то программа должна записать в выходном файле минимальное необходимое для этого количество ходов. Если черный король уже находится под шахом в исходном положении (даже если сама белая фигура находится под боем), программа должна вывести 0.

Если шах невозможен, то выходной файл должен быть пустым.

Пример: входной файл - “Be6 Ke2”. Выходной файл должен содержать “1”.